

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-236584

(43)Date of publication of application : 23.08.2002

(51)Int.Cl.

G06F 9/44

(21)Application number : 2001-031703

(71)Applicant : HITACHI LTD

(22)Date of filing : 08.02.2001

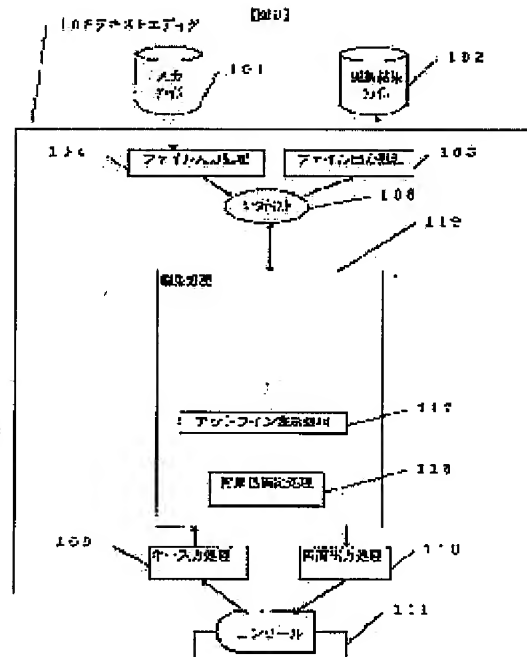
(72)Inventor : SAKANISHI SENICHIRO
KAWAI MASAO

(54) TEXT DISPLAY PROGRAM

(57)Abstract:

PROBLEM TO BE SOLVED: To efficiently display a source text which is developed according to a translation condition indicator as to a text editor for editing the source text including the condition translation indicator.

SOLUTION: A part in the source text which is not affected by a condition translation variable and a part which is affected by the condition translation variable are extracted from the source text respectively and for the extracted source text parts, different background colors are set. According to the depth of the nesting of condition translation indicators, indent displays are made by influence ranges of a translation conditional expression. The condition translation indicators are displayed as display items without fail and the ranges that the condition translation indicators affect are dynamically displayed or not displayed.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号
特開2002-236584
(P2002-236584A)

(43) 公開日 平成14年8月23日 (2002.8.23)

(51) Int.Cl.⁷
G 0 6 F 9/44

識別記号

F I
G 0 6 F 9/06

テーマコード* (参考)
6 2 0 B 5 B 0 7 6

審査請求 未請求 請求項の数 5 O L (全 21 頁)

(21) 出願番号 特願2001-31703(P2001-31703)

(22) 出願日 平成13年2月8日 (2001.2.8)

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 坂西 宣一郎

神奈川県横浜市戸塚区戸塚町5030番地 株

式会社日立製作所ソフトウェア事業部内

(72) 発明者 河井 政雄

神奈川県横浜市戸塚区戸塚町5030番地 株

式会社日立製作所ソフトウェア事業部内

(74) 代理人 100075096

弁護士 作田 康夫

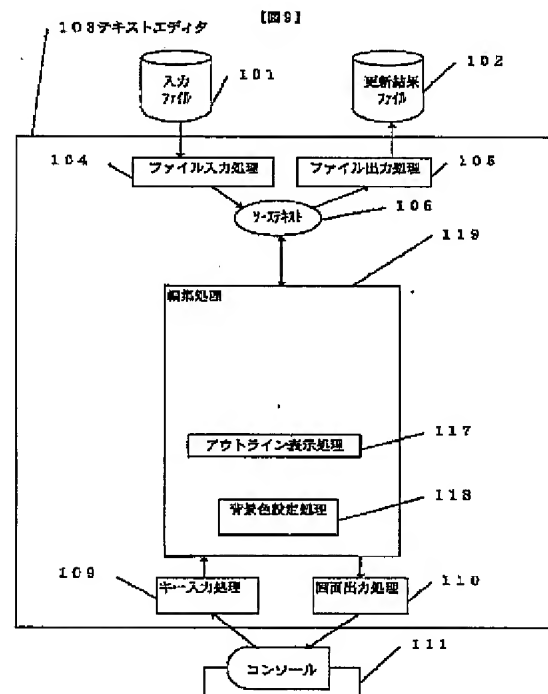
Fターム(参考) 5B076 AB15 DB04 DB05 DB06 EA02
EC02

(54) 【発明の名称】 テキスト表示プログラム

(57) 【要約】

【課題】 条件翻訳指示子を含むソーステキストを編集するテキストエディタに関し、翻訳条件指示子に従って展開されたソーステキストを効率よく表示する。

【解決手段】 ソーステキスト中の条件翻訳変数に影響されない部分と、条件翻訳変数に影響される部分とを、それぞれソーステキストから抽出し、抽出した各ソーステキスト部分をそれぞれ別々の色で背景色を設定する処理する。さらに条件翻訳指示子の入れ子の深さに応じて、前記翻訳条件式の影響範囲ごとにインデント表示をおこなう。また、条件翻訳指示子を表示項目として必ず表示し、前記条件翻訳指示子の影響する範囲を動的に表示・非表示する。



【特許請求の範囲】

【請求項 1】条件翻訳指示子を含むプログラムのソーステキストを表示するテキスト表示プログラムにおいて、ソーステキスト中の条件翻訳指示子の翻訳条件式ごとに影響範囲を認識し、ソーステキストを表示する際に、前記翻訳条件式の影響範囲と前記翻訳条件式の影響しない範囲で表示色の異なる文字表示をおこなうことを特徴とするテキスト表示プログラム。

【請求項 2】請求項 1 において、さらに条件翻訳指示子の入れ子の深さに応じて、前記翻訳条件式の影響範囲ごとにインデント表示をおこなうことを特徴とするテキスト表示プログラム。

【請求項 3】条件翻訳指示子を含むプログラムのソーステキストを表示するテキスト表示プログラムにおいて、ソーステキスト中の条件翻訳指示子ごとに影響範囲を抽出し、前記影響範囲の条件翻訳指示子を含む行を見出し項目として表示し、前記影響範囲の条件翻訳指示子を除く行の表示・非表示を動的におこなうことを特徴とするテキスト表示プログラム。

【請求項 4】条件翻訳指示子の翻訳条件式を満たす範囲にあっては、前記影響範囲の条件翻訳指示子を除く行の表示をおこなうことを特徴とする請求項 3 に記載のテキスト表示プログラム。

【請求項 5】条件翻訳指示子を含むプログラムのソーステキストを表示するテキスト表示プログラムにおいて、ソーステキストから条件翻訳指示子の翻訳変数と設定値を条件翻訳変数テーブルに抽出し、前記条件翻訳変数テーブルの内容の変更に応じて翻訳条件の再判定を行い、ソーステキストの翻訳条件を満たす範囲の再表示をおこなうことを特徴とする請求項 3 または請求項 4 に記載のテキスト表示プログラム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、プログラム言語のソーステキストファイルを表示あるいは編集するテキストエディタに関し、とりわけ条件翻訳指示子を含むプログラムのソーステキストを表示するのに好適なテキスト表示プログラムに関する。

【0002】

【従来の技術】プログラム言語には、コンパイラがソーステキスト上のどの部分を読み込むかをコンパイラに指示する条件翻訳指示子が規定されている場合が多い。例えば C 言語では、`#if` 条件翻訳指示子、`#else` 条件翻訳指示子、`#endif` 条件翻訳指示子が規定されていて、`#if` 条件翻訳指示子に指定された翻訳条件が成立する場合、コンパイラは `#if` 条件翻訳指示子以降 `#else` 条件翻訳指示子までの範囲のソーステキストを読み込み翻訳する。また `#if` 条件翻訳指示子に指定された翻訳条件が成立しない場合、コンパイラは `#else` 条件翻訳指示子以降 `#endif` 条件翻訳指示子までの範囲のソーステキストを読み込み翻

訳する。さらに C 言語では条件翻訳変数を定義する `#define` 条件翻訳指示子が規定されていて、`#if` 条件翻訳指示子の翻訳条件中でこの条件翻訳変数を使用することもできる。図 20 に、以降の本発明の説明において使用する条件翻訳指示子の文法と意味をまとめて示す。

【0003】また、以降の本発明の説明において、互いに背反関係にある (a) `#if` 条件翻訳指示子から `#else` 条件翻訳指示子で囲まれた範囲のソーステキスト部分、

(b) `#else` 条件翻訳指示子から `#endif` 条件翻訳指示子で囲まれた範囲のソーステキスト部分を、翻訳条件に応じてそれぞれ翻訳条件が成立する条件翻訳領域、又は翻訳条件が成立しない条件翻訳領域と表現して区別する。

【0004】従来は、第 2990701 号特許に記載のように、条件翻訳指示子が含まれるソーステキストファイルを読み込む場合、入力ファイル読み込み時に翻訳条件が成立する条件翻訳領域を更新ワークファイルに出力することで、翻訳条件が成立する条件翻訳領域だけを画面に表示するようになっていた。このため、第 2990701 号特許では、条件翻訳領域が入れ子構造になっているような複雑な場合については、処理ができないか、もしくは処理できても何度も繰り返し操作する必要があるという問題があった。

【0005】また特開平 5-88876 号公報に記載のように、条件翻訳指示子が含まれるソーステキストファイルを読み込む場合、入力ファイルを本文、コメント部、条件翻訳部に区別し、条件翻訳部の翻訳条件を判定し、判定結果に従って条件翻訳部を表示するか否かを決定するようになっていた。しかしこの特開平 5-88876 号公報では、`#else` 条件翻訳指示子を指定した場合の処理方法について配慮されていないため、`#else` 条件翻訳指示子を含むソーステキストや入れ子構造の条件翻訳領域を含む複雑なソーステキストを効率的に処理することができないという問題があった。

【0006】更に特開平 6-83602 号公報に記載のように、条件翻訳指示子が含まれるソーステキストファイルを読み込む場合、入力ファイル読み込み時に、プログラムの実行に関与するソーステキスト部分と、プログラムの実行に関与しないソーステキスト部分に分割してバッファに格納し、プログラムの実行に関与するソーステキスト部分のみを画面に表示するようになっていた。しかしこの特開平 6-83602 号公報では、プログラムの実行に関与する部分、即ち条件翻訳変数に影響されない共通なソーステキスト部分および翻訳条件が成立する条件翻訳領域において、各領域の境界が不明確で条件翻訳変数に影響されない共通なソーステキスト部分が容易に識別できないため、例えば図 2 に示すように、共通なソーステキスト部分と非共通なソーステキスト部分を同一部分であると誤認することによる修正ミスが発生しやすいという問題があった。

【0007】図 2 において、この従来技術によるテキス

トエディタで入力ファイル 10 を読み込んだ場合の表示内容が 11 である。ここでテキストエディタ使用者は、条件翻訳変数に影響されない共通なソーステキスト部分と翻訳条件が成立する条件翻訳領域の境界が不明確なため、式 15 が条件翻訳変数に影響されない共通なソーステキスト部分であり、式 16 が翻訳条件が成立する条件翻訳領域であることを認識できず、式 15 と式 16 は同一部分であり冗長であると誤って判断してしまい、式 15 を変更し式 16 を削除して、最終的に 12 に示す内容に修正してしまう。ここで編集後のソーステキストをファイルに出力した場合の内容が 13 である。13 では、条件翻訳変数に影響されない共通なソーステキスト部分であり本来変更してはならない式 15 が式 17 に修正されている。その結果、条件翻訳指示子「#if PC = 1」が成立しない場合、変数「a」の値が 10 と 13 では異なるため、誤動作する可能性がある。

【0008】

【発明が解決しようとする課題】以上で説明した従来技術では（１）条件翻訳領域が入れ子構造になっているような複雑な条件翻訳があるソーステキストの場合において、翻訳条件に従って展開されたソーステキストを効率よく表示することができない、（２）条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすい、という問題があった。

【0009】本発明の目的は、上記従来技術の問題点に対して、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳があるプログラムのソーステキストの場合において、翻訳条件に従って展開されたソーステキストを効率よく表示するテキストエディタを提供することにある、また、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できるようにし、共通なソーステキスト部分に対する修正ミスが起りにくいテキストエディタを提供することにある。

【0010】

【課題を解決するための手段】上記目的は、ソーステキスト中から各条件翻訳領域を抽出し、条件翻訳指示子の翻訳条件式ごとに影響範囲を認識し、ソーステキストを表示する際に、前記翻訳条件式に影響範囲と前記翻訳条件式に影響されない範囲を視覚的に異なる表示をおこなうようにすることで達成される。例えば、前記範囲のテキストを異なる色分け表示をおこなうようにする。

【0011】さらに、前記翻訳条件式の影響範囲が、複数の条件翻訳領域の入れ子構造になっている場合には、入れ子の深さに応じて翻訳条件式の影響範囲をインデント表示するようにする。より詳細には、複数の翻訳条件

式の影響範囲が入れ子の構造になっている場合、入れ子になっている内側の翻訳条件の影響範囲のソーステキストと該影響範囲の外側のソーステキストをインデント表示をするとともに異なる表示色でテキスト表示するようにする。

【0012】さらに、上記条件翻訳領域の表示をおこなう際に、翻訳条件式が成立する条件翻訳領域のみ表示し、翻訳条件式が成立しない条件翻訳領域は表示しないようにしてもよい。

【0013】また、条件翻訳指示子を境界としてソーステキストを複数の領域に分割するとともに、各領域の境界に相当する条件翻訳指示子は必ず表示し、分割後の各領域を、条件翻訳が成立するか否かという条件、もしくはテキストエディタ使用者の指示に従って動的に表示・非表示を制御するようにする。

【0014】さらに、ソーステキストから条件翻訳指示子の翻訳変数と設定値を読み出し、プログラムの条件翻訳変数テーブルに格納し、該条件翻訳テーブルを表示・変更可能とし、前記テーブルに基づき、上記のとおりソーステキストを表示するようにする。

【0015】

【発明の実施の形態】以下、本発明によるテキストエディタについて図をもちいて詳細に説明する。

【0016】（実施例 1）本実施例では、ソーステキスト中の条件翻訳指示子の翻訳条件を判定し、翻訳条件が成立する条件翻訳領域を条件翻訳指示子に対応したそれぞれ別々の色で背景色を色分けし、かつ条件翻訳領域の入れ子の深さに応じてインデント表示し、翻訳条件が成立しない条件翻訳領域を表示しないことによって、翻訳条件に従って展開されたソーステキストを効率よく表示するとともに、ソーステキスト中の条件翻訳変数に影響されない共通なソーステキスト部分をテキストエディタ使用者が容易に識別できるように、前述の条件翻訳領域と区別可能な色で背景色を色分け表示する。更に、各条件翻訳変数に影響されるソーステキスト部分において、互いに背反関係にある、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通なソーステキスト部分を、テキストエディタ使用者が容易に識別できるように、前述の条件翻訳領域他と区別可能な色で背景色を色分け表示する。

【0017】なお、以下の説明において、条件翻訳変数に影響されない共通なソーステキスト部分を共通ソーステキスト部分 1 と表現し、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通なソーステキスト部分を共通ソーステキスト部分 2 と表現して区別する。

【0018】図 1 は、本発明の第一の実施例を実現するテキストエディタの構成例を示すブロック図である。図 1 において、テキストエディタ 103 は、プログラム言語のソーステキストを内容とする入力ファイル 101 を

外部記憶装置から読み込みソーステキスト106としてテキストエディタ103内部に保持するファイル入力処理104と、コンソール111からのテキストエディタ使用者の操作を入力するキー入力手段109と、前記操作に基づいてソーステキスト106を編集する編集処理107と、この編集結果によって更新されたソーステキスト106をコンソール111に表示する画面出力処理110と、前記編集結果によって更新されたソーステキスト106を外部記憶装置の更新結果ファイル102に出力するファイル出力処理105から構成される。

【0019】編集処理107は、さらに、ソーステキスト106中の条件翻訳領域に対して、翻訳条件が成立する条件翻訳領域をコンソール111に表示する条件成立翻訳領域表示処理113と、ソーステキスト106中の共通ソーステキスト部分1および共通ソーステキスト部分2と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定する背景色設定処理118から構成される。

【0020】図3は、以下の説明において、本発明によるテキストエディタが読み込む入力ファイルの例である。

【0021】図4は、図3のソーステキストを図1の構成から成るテキストエディタで読み込んだ場合の、コンソール111の表示例である。

【0022】図4において、コンソール111の表示画面には、入力ファイル101から読み込んだソーステキスト106を表示するソーステキストウィンドウ301を表示し、テキストエディタ使用者は、このソーステキストウィンドウ301に対して編集作業を行う。

【0023】図5は、図4のコンソール111に表示されているソーステキストウィンドウ301の表示内容の詳細例である。

【0024】図5では、ソーステキストウィンドウ301に、条件翻訳変数に影響されない共通ソーステキスト部分1と、翻訳条件が成立する条件翻訳領域を表示する。

【0025】図5において、401はソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分1であり、テキストエディタ使用者が共通ソーステキスト部分1を容易に識別できるように、背景色設定処理118により指定された色で背景色を色分け表示する。402は、各条件翻訳変数に影響されるソーステキスト部分において、互いに背反関係にある、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通な共通ソーステキスト部分2であり、テキストエディタ使用者が共通ソーステキスト部分2を容易に識別できるように、背景色設定処理118により指定された色で背景色を色分け表示する。403は条件翻訳変数「PC」に影響されるソーステキスト部分であり、404は条件翻訳変数「WS」に影響されるソーステキス

ト部分であり、405は条件翻訳変数「MACHINE1」に影響されるソーステキスト部分であり、テキストエディタ使用者が条件翻訳変数に影響されない共通ソーステキスト部分1である401と容易に区別できるように、かつそれぞれの条件翻訳変数に影響されるソーステキスト部分を互いに区別できるように、背景色設定処理118によりそれぞれ指定された別々の色で背景色を色分け表示するとともに、条件翻訳指示子の入れ子の深さに応じてインデント表示する。

10 【0026】図3に示した入力ファイル例では、条件翻訳変数「MACHINE1」に影響されるソーステキスト部分は、条件翻訳変数「WS」に影響されるソーステキスト部分の入れ子となっており、更にこの条件翻訳変数「WS」に影響されるソーステキスト部分は、条件翻訳変数「PC」に影響されるソーステキスト部分の入れ子となる構造となっている。このように条件翻訳変数に影響される部分が入れ子構造になっているソーステキストの場合、図5に示すように本実施例では、入れ子の深さに応じてインデント表示するとともに、各条件翻訳変数毎に背景の色を定め、更に入れ子の深さが一番深い部分が一番手前になるよう重ねて表示する。この結果、条件翻訳領域に影響されるソーステキスト部分が入れ子構造になっていた場合でも、どの条件翻訳変数に影響を受けるのかを背景色で明示しながらソーステキストを表示することができる。

20 【0027】図5において、406、407および408は、各条件翻訳変数に影響されるソーステキスト部分のインデント領域であり、条件翻訳変数に影響されるソーステキスト部分が入れ子構造になっている場合、入れ子階層を明確化するとともに、テキストエディタ使用者がソーステキストがどの入れ子階層に属するか容易に識別できるように、各条件翻訳変数毎に定めた背景色で塗りつぶす。

30 【0028】なお図5では、401、402、403、404及び405を枠で囲んで表現しているが、それぞれテキストエディタ使用者によって指示された色で背景色を表示するだけでもよい。更に図5では、401、402、403、404及び405の背景色を色分け表示しているが、背景色ではなく文字色を色分け表示してもよい。

40 【0029】また図5では、各条件翻訳領域をインデントして表示しているが、条件翻訳領域の入れ子が深くなった場合にソーステキストの連続性が失われるという点に配慮し、テキストエディタ使用者からの指示によりソーステキストの文法に従ったインデントで表示することもできる。

【0030】次に、本実施例によるテキストエディタの処理手順をフローチャートを用いて説明する。

50 【0031】図6は、図1の構成から成るテキストエディタの処理の流れを説明するフローチャートである。

【0032】図6では、入力ファイル101からソーステキスト106を読み込み（ステップ501）、ソーステキスト106中の共通ソーステキスト部分1および共通ソーステキスト部分2と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定し（ステップ502。本処理の詳細は後述する図7で説明する。なおフローチャート中で縦線が二重になっている箱は、別の処理を呼び出すことを示す）、条件翻訳変数に影響されない共通ソーステキスト部分1と翻訳条件が成立する条件翻訳領域をコンソール111に表示する（ステップ503。本処理の詳細は後述する図8で説明する）。

【0033】次にテキストエディタ使用者からの編集操作を受け取ったキー入力処理109からの指示に従い、ソーステキスト106に対して編集を行い（ステップ504）、編集操作によって更新されたソーステキスト106をコンソール111に表示する（ステップ505）。

【0034】ここでテキストエディタ使用者がテキストエディタ終了を指示したか判定し（ステップ506）、終了を指示した場合はソーステキスト106を更新結果ファイル102に出力し（ステップ507）、テキストエディタを終了する。

【0035】テキストエディタ終了が指示されていない場合、テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行ったか判定し（ステップ508）、#define条件翻訳指示子の変更または新規追加を行っていない場合、ステップ504に戻り以降のステップを繰り返す。

【0036】テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行った場合、ステップ502に戻り、共通ソーステキスト部分1および共通ソーステキスト部分2と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ503で条件翻訳変数に影響されない共通ソーステキスト部分1と翻訳条件が成立する条件翻訳領域をコンソール111に再表示する。そしてステップ504以降のステップを繰り返す。

【0037】図7は、図6におけるステップ502で実行される、背景色設定処理118の処理の流れを説明するフローチャートである。

【0038】図7では、ソーステキスト106からソーステキストを1行取り出し（ステップ701）、ソーステキスト終端か判定し（ステップ702）、ソーステキスト終端の場合は呼び出し元に処理を戻す。

【0039】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#if」であるか判定し（ステップ703）、行頭が「#if」以外の場合は、ソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分1としてテキストエディタ使用者に指定された色で背景色を色分け表示するよう取り出した行に

設定し（ステップ704）、ステップ701に戻り以降のステップを繰り返す。

【0040】取り出したソーステキスト行の行頭が「#if」の場合、ソーステキスト106から行頭が「#else」であるソーステキスト行までの範囲を取り出し退避し（ステップ705）、次にソーステキスト行を取り出す位置を「#else」の次の行に設定し（ステップ706）、ソーステキスト106から行頭が「#endif」であるソーステキスト行までの範囲を取り出し退避し（ステップ707）、次にソーステキスト行を取り出す位置を「#endif」の次の行に設定する（ステップ708）。

【0041】次にステップ705およびステップ707で退避した2つのソーステキストに対して、対応する条件翻訳変数に影響されるソーステキスト部分としてテキストエディタ使用者に指定された色で背景色を色分け表示するよう設定する（ステップ709）。

【0042】次にステップ705およびステップ707で退避した2つのソーステキストを比較し、双方に共通なソーステキスト部分とそれ以外のソーステキスト部分に分割し（ステップ710）、分割した共通なソーステキスト部分に対して、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方に共通な共通ソーステキスト部分2としてテキストエディタ使用者に指定された色で背景色を色分け表示するよう設定する（ステップ711）。

【0043】次にステップ705およびステップ707で退避した2つのソーステキストに対して本処理を再帰的に実行し（ステップ712）、次にソーステキスト行を取り出す位置をステップ708で設定した位置に再設定し（ステップ713）、ステップ701に戻り以降のステップを繰り返す。

【0044】図8は、図6におけるステップ503で実行される、条件成立翻訳領域表示処理113の処理の流れを説明するフローチャートである。

【0045】図8では、ソーステキスト106からソーステキストを1行取り出し（ステップ801）、ソーステキスト終端か判定し（ステップ802）、ソーステキスト終端の場合は呼び出し元に処理を戻す。

【0046】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#if」であるか判定し（ステップ803）、行頭が「#if」以外の場合は、取り出したソーステキスト行をコンソール111に表示するよう画面出力処理110に指示し（ステップ804）、ステップ801に戻り以降のステップを繰り返す。

【0047】取り出したソーステキスト行の行頭が「#if」の場合、#if条件翻訳指示子から翻訳条件を抽出し（ステップ805）、ソーステキスト106から行頭が「#else」であるソーステキスト行までの範囲を取り出し退避し（ステップ806）、次にソーステキスト行を

取り出す位置を「#else」の次の行に設定し（ステップ 807）、ソーステキスト 106 から行頭が「#endif」であるソーステキスト行までの範囲を取り出し退避し（ステップ 808）、次にソーステキスト行を取り出す位置を「#endif」の次の行に設定する（ステップ 809）。

【0048】次にステップ 805 で抽出した翻訳条件を判定する（ステップ 810）。翻訳条件が成立する場合、ステップ 806 で退避した、#if 条件翻訳指示子から #else 条件翻訳指示子までの範囲のソーステキストを
10 コンソール 111 に条件翻訳領域の入れ子の深さに応じてインデントして表示するよう画面出力処理 110 に指示する（ステップ 811）。ステップ 805 で抽出した翻訳条件が成立しない場合、ステップ 808 で退避した、#else 条件翻訳指示子から #endif 条件翻訳指示子までの範囲のソーステキストをコンソール 111 に条件翻訳領域の入れ子の深さに応じてインデントして表示するよう画面出力処理 110 に指示する（ステップ 812）。

【0049】次にステップ 806 及びステップ 808 で退避したソーステキストに対して本処理を再帰的に実行し（ステップ 813）、次にソーステキスト行を取り出す位置をステップ 809 で設定した位置に再設定し（ステップ 814）、ステップ 801 に戻り以降のステップを繰り返す。

【0050】なお図 8 の各ステップにおいて画面出力を行う場合は、図 6 のステップ 502 によって設定された色で背景色を色分け表示する。

【0051】以上で説明した実施例によって、従来技術で問題であった、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳を含むソーステキストの場合に、翻訳条件に従って展開されたソーステキストを効率よく表示することができないという問題を軽減することができる。また、従来技術で問題であった、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすいという問題を軽減することもできる。更に本実施例によれば、ソーステキスト中の翻訳条件が成立する条件翻訳領域と
40 翻訳条件が成立しない条件翻訳領域の双方に共通なソーステキスト部分が存在する場合、このソーステキスト部分は翻訳条件の判定結果にかかわらず常に実行されるため、条件翻訳領域内に存在する必要はなく条件翻訳領域の外に移動すべきであることを、テキストエディタ使用者が容易に判断できるという効果を持つ。

【0052】なお、以上で説明した実施例では、ソーステキスト中の条件翻訳指示子の翻訳条件を判定し、翻訳条件が成立する条件翻訳領域だけを表示するという表示方法であるが、本実施例における条件翻訳指示子の翻訳

条件の判定および判定結果に基づいて条件翻訳領域の表示の有無を決定する処理を実行せず、更には条件翻訳領域以下をインデントして全て表示することによって、ソーステキスト全てを表示することができ、かつソーステキスト中の共通ソーステキスト部分 1 および共通ソーステキスト部分 2 と各条件翻訳変数に影響される各ソーステキスト部分をそれぞれ別々の色で背景色を色分け表示するようにできることは言うまでもない。この方法と本実施例で説明した方法を選択するように構成することによって、テキストエディタ使用者は、翻訳条件が成立する条件翻訳領域だけを表示するかソーステキスト全てを表示するか、必要に応じて切替えることができる。

【0053】本実施例で、ソーステキストの区別を背景色の変更によりおこなう例を説明したが、文字色の変更や、文字種別により区別してもよいことは言うまでもない。

【0054】（実施例 2）次に、本発明の第二の実施例によるテキストエディタについて説明する。

【0055】本発明の第二の実施例は、第一の実施例とは別の手段によって従来技術の問題点を解決することを目的とした実施例である。

【0056】本実施例では、条件翻訳指示子を境界としてソーステキストを複数の領域に分割するとともに、各領域の境界に相当する条件翻訳指示子は必ず表示し、分割後の各領域を、テキストエディタ使用者の指示に従って動的に表示・非表示を制御する。本実施例ではこれをアウトライン表示と表現し、前記の分割した領域をアウトライン表示領域と表現する。

【0057】図 9 は、本発明の第二の実施例を実現するテキストエディタの構成例を示すブロック図である。

【0058】図 9 において、編集処理 119 を除いて、その構成は図 1 の実施例 1 の構成と同様であり、同一のブロックは同一の記号で示している。

【0059】図 9 における編集処理 119 は、各条件翻訳領域のアウトライン表示を行うアウトライン表示処理 117 と、ソーステキスト 106 中の共通ソーステキスト部分 1 および共通ソーステキスト部分 2 と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定する背景色設定処理 118 から構成される。本実施例の場合、図 3 のソーステキストを図 9 の構成から成るテキストエディタで読み込んだ場合のコンソール 111 の表示方法は、図 4 に示す本発明の第一の実施例の場合と同様であるため、説明は省略する。

【0060】図 10 は、図 3 のソーステキストを図 9 の構成から成るテキストエディタで読み込んだ場合の、ソーステキストウィンドウ 301 の表示例の詳細である。

【0061】本実施例の中核部分は、#if、#else および #endif の各条件翻訳指示子を、入れ子の深さに応じてインデント処理して全て表示するとともに、これら条件翻訳指示子によって囲まれたソーステキストについては、

各条件翻訳指示子毎にアウトライン表示領域展開状態指示子を受け、その指示に従ってソーステキストを表示するか否かの制御をすることにある。ここで、アウトライン表示領域展開状態指示子への設定は、#if条件翻訳指示子の翻訳条件が成立する条件翻訳領域は表示をし、逆に翻訳条件が成立しない条件翻訳領域は表示をしないという状態を初期状態とし、以降はテキストエディタ使用者の指示によって表示・非表示を制御するという方法である。

【0062】図10において、401はソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分1であり、テキストエディタ使用者が共通ソーステキスト部分1を容易に識別できるように、背景色設定処理118により指定された色で背景色を色分け表示する。402は、各条件翻訳変数に影響されるソーステキスト部分において、互いに背反関係にある、翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方で共通な共通ソーステキスト部分2であり、テキストエディタ使用者が共通ソーステキスト部分2を容易に識別できるように、背景色設定処理118により指定された色で背景色を色分け表示する。409は条件翻訳指示子「#if PC == 1」に対応する#else条件翻訳指示子と#endif条件翻訳指示子を境界とする展開状態のアウトライン表示領域であり、条件翻訳領域が入れ子構造になっている場合、このアウトライン表示領域の中に、入れ子の条件翻訳領域に対応するアウトライン表示領域がインデント処理されて表示される。410は条件翻訳指示子「#if WS == 1」と#else条件翻訳指示子を境界とする展開状態のアウトライン表示領域であり、アウトライン表示領域409の中に包含されている。411は条件翻訳指示子「#if MACHINE1 == 1」と#else条件翻訳指示子を境界とする展開状態のアウトライン表示領域であり、アウトライン表示領域410の中に包含されている。412は展開されたアウトライン表示領域を記号「+」で表現し、展開されていないアウトライン表示領域を記号「-」で表現するアウトライン表示領域展開状態指示子であり、テキストエディタ使用者は、このアウトライン表示領域展開状態指示子412に対して操作を行うことにより、アウトライン表示領域の展開・非展開状態を変更することができる。

【0063】なお本実施例では、テキストエディタの起動直後の各アウトライン表示領域の展開・非展開状態は、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域は展開状態で、翻訳条件が成立しない条件翻訳領域に対応するアウトライン表示領域は非展開状態となるように設定する。図10はこのような設定で表示した例であり、翻訳条件が成立する条件翻訳領域に対応する、条件翻訳指示子「#if PC == 1」に対応する#else条件翻訳指示子と#endif条件翻訳指示子を境界とするアウトライン表示領域409と、条件翻訳指示子「#if

WS == 1」と#else条件翻訳指示子を境界とするアウトライン表示領域410と、条件翻訳指示子「#if MACHINE1 == 1」と#else条件翻訳指示子を境界とするアウトライン表示領域411を展開状態として表示し、その他の翻訳条件が成立しない条件翻訳領域に対応するアウトライン表示領域はすべて非展開状態で表示している。

【0064】また本実施例では、各アウトライン表示領域の展開・非展開状態をテキストエディタ使用者が変更しても、テキストエディタ起動直後の状態である、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域だけを展開状態にするように戻すこともできる。

【0065】なお図10では、401及び402を枠で囲んで表現しているが、テキストエディタ使用者によって指示された色で背景色を表示するだけでもよい。更に図10では、401及び402の背景色を色分け表示しているが、背景色ではなく文字の色を色分け表示してもよい。

【0066】図11は、図10における条件翻訳指示子「#if PC == 1」と#else条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を、例えば、「#if PC == 1」の行頭にあるアウトライン表示領域展開状態指示子412をテキストエディタ使用者がマウスでダブルクリックするというような操作を行うことで、展開させた場合の表示例である。

【0067】図11において、条件翻訳指示子「#if PC == 1」の行頭にあるアウトライン表示領域展開状態指示子412の記号が、非展開状態を示す「+」から、展開状態を示す「-」に変更されているとともに、図10では表示されていなかったアウトライン表示領域413が展開状態で表示されている。

【0068】以下、本実施例によるテキストエディタの処理手順をフローチャートを用いて説明する。

【0069】図12は、図9の構成から成るテキストエディタの処理の流れを説明するフローチャートである。

【0070】図12では、入力ファイル101からソーステキスト106を読み込み（ステップ1401）、ソーステキスト106中の共通ソーステキスト部分1および共通ソーステキスト部分2と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定し（ステップ1402。なお本処理は実施例1の図6におけるステップ502と同様であり、詳細は図7を参照）、ソーステキスト106をコンソール111にアウトライン表示する（ステップ1403。本処理の詳細は後述する図13で説明する）。

【0071】次にテキストエディタ使用者からの編集操作を受け取ったキー入力処理109からの指示に従い、ソーステキスト106に対して編集を行い（ステップ1404）、編集操作によって更新されたソーステキスト106をコンソール111に表示する（ステップ1405）。

【0072】ここでテキストエディタ使用者がテキストエディタ終了を指示したか判定し（ステップ1406）、終了を指示した場合はソーステキスト106を更新結果ファイル102に出力し（ステップ1407）、テキストエディタを終了する。

【0073】テキストエディタ終了が指示されていない場合、テキストエディタ使用者が任意のアウトライン表示領域に対して展開・非展開状態の変更操作を行ったか判定し（ステップ1408）、変更操作を行った場合、対応するアウトライン表示領域の展開・非展開状態を変更し、変更したアウトライン表示領域をコンソール111に再表示し（ステップ1409）、ステップ1404に戻り以降のステップを繰り返す。

【0074】テキストエディタ使用者がアウトライン表示領域の展開・非展開状態の変更操作を行っていない場合、テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行ったか判定し（ステップ1410）、変更または新規追加を行った場合、ステップ1402に戻り、共通ソーステキスト部分1および共通ソーステキスト部分2と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ1403以降のステップを繰り返す。

【0075】テキストエディタ使用者が#define条件翻訳指示子の変更または新規追加を行っていない場合、テキストエディタ使用者が、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域だけを展開表示するよう指示したか判定し（ステップ1411）、指示していない場合、ステップ1404に戻り以降のステップを繰り返す。

【0076】テキストエディタ使用者が、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域だけを展開表示するよう指示した場合、ステップ1403に戻り、全てのアウトライン表示領域に対して、翻訳条件が成立する条件翻訳領域に対応するアウトライン表示領域を展開状態に設定し、翻訳条件が成立しない条件翻訳領域に対応するアウトライン表示領域を非展開状態に設定し、全てのアウトライン表示領域をコンソール111に再表示し、ステップ1404以降のステップを繰り返す。

【0077】図13は、図12におけるステップ1403で実行される、アウトライン表示処理117の処理の流れを説明するフローチャートである。

【0078】図13では、ソーステキスト106からソーステキストを1行取り出し（ステップ1501）、ソーステキスト終端が判定し（ステップ1502）、ソーステキスト終端の場合は呼び出し元に処理を戻す。

【0079】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#if」であるか判定し（ステップ1503）、行頭が「#if」以外の場合は、取り出したソーステキスト行をコンソール111に表示

するよう画面出力処理110に指示し（ステップ1504）、ステップ1501に戻り以降のステップを繰り返す。

【0080】取り出したソーステキスト行の行頭が「#if」の場合、#if条件翻訳指示子から翻訳条件を抽出し（ステップ1505）、ソーステキスト106から行頭が「#else」であるソーステキスト行までの範囲を取り出し退避し、#if条件翻訳指示子と#else条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を作成し（ステップ1506）、次にソーステキスト行を取り出す位置を「#else」の次の行に設定し（ステップ1507）、ソーステキスト106から行頭が「#endif」であるソーステキスト行までの範囲を取り出し退避し、#else条件翻訳指示子と#endif条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を作成し（ステップ1508）、次にソーステキスト行を取り出す位置を「#endif」の次の行に設定する（ステップ1509）。

【0081】次にステップ1505で抽出した翻訳条件を判定する（ステップ1510）。翻訳条件が成立する場合、ステップ1506で作成した、#if条件翻訳指示子と#else条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を展開状態に設定する（ステップ1511）。翻訳条件が成立しない場合、ステップ1508で作成した、#else条件翻訳指示子と#endif条件翻訳指示子を境界とする非展開状態のアウトライン表示領域を展開状態に設定する（ステップ1512）。

【0082】次にステップ1506およびステップ1508で作成した2つのアウトライン表示領域をコンソール111に条件翻訳領域の入れ子の深さに応じてインデントして表示するよう画面出力処理110に指示し（ステップ1513）、ステップ1506およびステップ1508で退避したソーステキストに対して本処理を再帰的に実行し（ステップ1514）、次にソーステキスト行を取り出す位置をステップ1509で設定した位置に再設定し（ステップ1515）、ステップ1501に戻り以降の処理を繰り返す。

【0083】なお図13の各ステップにおいて画面出力を行う場合は、図12のステップ1402によって設定された色で背景色を色分け表示する。

【0084】以上で説明した実施例によって、従来技術で問題であった、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳を含むソーステキストの場合に、翻訳条件に従って展開されたソーステキストを効率よく表示することができないという問題を軽減することができる。また、従来技術で問題であった、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中で条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすいという問題を軽減することもできる。更に本実施例によれば、

条件翻訳領域をテキストエディタ使用者の指示に従って動的に表示・非表示を制御することができるという効果と、ソーステキスト中の翻訳条件が成立する条件翻訳領域と翻訳条件が成立しない条件翻訳領域の双方に共通なソーステキスト部分が存在する場合、このソーステキスト部分は翻訳条件の判定結果にかかわらず常に実行されるため、条件翻訳領域内に存在する必要はなく条件翻訳領域の外に移動すべきであることを、テキストエディタ使用者が容易に判断できるという効果を持つ。

【0085】なお、以上で説明した実施例では、ソーステキスト中の共通ソーステキスト部分1および共通ソーステキスト部分2と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を色分け表示するという表示方法であるが、本実施例における背景色設定処理を実行しないことによって、それぞれを色分け表示せずに、各条件翻訳領域のアウトライン表示だけにすることもできる。このような方法においても、条件翻訳変数に影響されない共通ソーステキスト部分1と各条件翻訳変数に影響されるソーステキスト部分のそれぞれがアウトライン表示によって明確化されるため、従来技術の問題点を解決することができる。

【0086】（実施例3）次に、本発明の第三の実施例によるテキストエディタについて説明する。

【0087】本発明の第三の実施例は、本発明の第一の実施例により表示したソーステキストに対して、テキストエディタ使用者が、条件翻訳変数を容易に変更する手段を提供することで、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できるようにする実施例である。

【0088】本実施例では、ソーステキスト中の#define条件翻訳指示子から条件翻訳変数の名称と値を抽出した条件翻訳変数テーブルを作成し、作成した条件翻訳変数テーブルの内容をソーステキストを表示するウィンドウとは別のウィンドウに表示し、テキストエディタ使用者がこのウィンドウに表示された条件翻訳変数テーブルの内容を参照・変更すると、この変更内容をソーステキスト中の対応する#define条件翻訳指示子に反映し、条件翻訳指示子の翻訳条件の再判定を行い、翻訳条件を反映したソーステキストを表示する。

【0089】図14は、本発明の第三の実施例を実現するテキストエディタの構成例を示すブロック図である。

【0090】図14において、編集処理120を除いて、その構成は図1の実施例1の構成と同様であり、同一ブロックは同一の記号で示している。

【0091】図14における編集処理120は、図1の編集処理107に対して、ソーステキスト106中の条件翻訳変数を抽出し条件翻訳変数テーブル108を作成する条件翻訳変数テーブル作成処理114と、条件翻訳変数テーブル108の内容を表示し、テキストエディタ使用者からの操作を受け取り、ソーステキスト106の

表示内容と条件翻訳変数テーブル108の内容を更新する条件翻訳変数テーブル表示制御処理115を追加した構成である。

【0092】図15は、図3のソーステキストを図14の構成から成るテキストエディタで読み込んだ場合のコンソール111の表示例である。

【0093】図15において、コンソール111の表示画面には、入力ファイル101から読み込んだソーステキスト106を表示するソーステキストウィンドウ301と、条件翻訳変数テーブル108の内容を表示した条件翻訳変数ウィンドウ302を表示する。

【0094】本実施例の場合、図15のコンソール111に表示されているソーステキストウィンドウ301の表示内容の詳細は、図5に示す本発明の第一の実施例の場合と同様であるため、説明は省略する。

【0095】図16は、図15のコンソール111に表示されている条件翻訳変数ウィンドウ302の表示内容の詳細例である。なお、図16の内容は、図14における条件翻訳変数テーブル108の内容と同じである。

【0096】テキストエディタ使用者は、ソーステキストウィンドウ301に対して編集作業を行うとともに、条件翻訳変数ウィンドウ302に対して条件翻訳変数の名称や値の変更操作を行うことができる。テキストエディタ使用者が条件翻訳変数ウィンドウ302に対して変更操作を行った場合、変更操作を条件翻訳変数テーブル108に反映するとともに、ソーステキスト106中の対応する#define条件翻訳指示子に反映し、ソーステキストウィンドウ301の表示内容が更新される。またテキストエディタ使用者がソーステキストウィンドウ301に表示されている#define条件翻訳指示子を追加・変更した場合、変更内容を条件翻訳変数テーブル108に反映させ、条件翻訳変数ウィンドウ302の表示内容を更新する。更に、テキストエディタ使用者が条件翻訳変数ウィンドウ302に対して、ソーステキスト中の共通ソーステキスト部分1および共通ソーステキスト部分2、または各条件翻訳変数に影響されるソーステキスト部分の背景色を変更した場合、対応するソーステキスト部分を変更された背景色で再色分け表示する。

【0097】図16において、1801は条件翻訳変数テーブルの出現順の番号である。1802は条件翻訳変数の名称であり、1803は条件翻訳変数の値であり、テキストエディタ使用者はそれぞれの内容を変更することができる。1804は条件翻訳変数を定義しているソーステキスト行番号である。1805はこの条件翻訳変数に影響されるソーステキスト部分の背景色であり、テキストエディタ使用者は内容を変更することができる。

【0098】また図16の表中の項番4は、ソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分1を示す特別な項番であり、テキストエディタ使用者は、項番4の項目1805を変更することができ、

ソーステキスト中の条件翻訳変数に影響されない共通ソーステキスト部分 1 を、変更された背景色で再色分け表示する。同様に図 16 の表中の項番 5 は、ソーステキスト中の共通ソーステキスト部分 2 を示す特別な項番であり、テキストエディタ使用者は、項番 5 の項目 1805 を変更することができ、ソーステキスト中の共通ソーステキスト部分 2 を、変更された背景色で再色分け表示する。

【0099】以下、本実施例によるテキストエディタの処理手順をフローチャートを用いて説明する。図 17 は、図 14 の構成から成るテキストエディタの処理の流れを説明するフローチャートである。

【0100】図 17 では、入力ファイル 101 からソーステキスト 106 を読み込み（ステップ 1901）、ソーステキスト 106 中の共通ソーステキスト部分 1 および共通ソーステキスト部分 2 と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を設定し（ステップ 1902。なお本処理は実施例 1 の図 6 におけるステップ 502 と同様であり、詳細は図 7 を参照）、条件翻訳変数テーブル 108 を作成し（ステップ 1903。本処理の詳細は後述する図 18 で説明する）、ソーステキスト 106 中の条件翻訳変数に影響されない共通ソーステキスト部分 1 と、翻訳条件が成立する条件翻訳領域をコンソール 111 に表示し（ステップ 1904。なお本処理は実施例 1 の図 6 におけるステップ 503 と同様であり、詳細は図 8 を参照）、条件翻訳変数テーブル 108 の内容を条件翻訳変数ウィンドウ 302 に表示する（ステップ 1905）。

【0101】次にテキストエディタ使用者からの条件翻訳変数ウィンドウ 302 に対する操作を待つ（ステップ 1906）。

【0102】テキストエディタ使用者からの操作がテキストエディタ終了を指示しているか判定し（ステップ 1907）、終了を指示した場合はソーステキスト 106 を更新結果ファイル 102 に出力し（ステップ 1908）、テキストエディタを終了する。

【0103】テキストエディタ終了が指示されていない場合、テキストエディタ使用者が条件翻訳変数ウィンドウ 302 に対して変更操作を行ったか判定し（ステップ 1909）、変更操作を行った場合、変更内容に対応するソーステキスト 106 中の `#define` 条件翻訳指示子に反映させ（ステップ 1910）、条件翻訳変数ウィンドウ 302 に指定された背景色で色分け処理するように背景色設定処理 118 に指示し（ステップ 1911）、ステップ 1902 に戻り、ソーステキスト 106 中の共通ソーステキスト部分 1 および共通ソーステキスト部分 2 と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ 1903 で条件翻訳変数テーブルを再構築し、ステップ 1904 でソース

ーステキスト部分 1 と、翻訳条件が成立する条件翻訳領域をコンソール 111 に再表示し、ステップ 1905 で条件翻訳変数テーブル 108 の内容を条件翻訳変数ウィンドウ 302 に再表示し、以降のステップを繰り返す。

【0104】テキストエディタ使用者が条件翻訳変数ウィンドウ 302 に対して変更操作を行っていない場合、テキストエディタ使用者がソーステキスト 106 中の `#define` 条件翻訳指示子の変更または新規追加を行ったか判定し（ステップ 1912）、変更または新規追加を行った場合、ステップ 1902 に戻り、ソーステキスト 106 中の共通ソーステキスト部分 1 および共通ソーステキスト部分 2 と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ 1903 で条件翻訳変数テーブルを再構築し、ステップ 1904 でソーステキスト 106 中の条件翻訳変数に影響されない共通ソーステキスト部分 1 と、翻訳条件が成立する条件翻訳領域をコンソール 111 に再表示し、ステップ 1905 で条件翻訳変数テーブル 108 の内容を条件翻訳変数ウィンドウ 302 に再表示し、以降のステップを繰り返す。

【0105】テキストエディタ使用者が `#define` 条件翻訳指示子の変更または新規追加を行っていない場合、テキストエディタ使用者がソーステキスト 106 中の `#if`、`#else` および `#endif` の各条件翻訳指示子を追加・変更・削除したかどうか判定する（ステップ 1913）。各条件翻訳指示子を追加・変更・削除していない場合、ステップ 1906 に戻り以降のステップを繰り返す。

【0106】各条件翻訳指示子を追加・変更・削除した場合、ステップ 1902 に戻り、ソーステキスト 106 中の共通ソーステキスト部分 1 および共通ソーステキスト部分 2 と各条件翻訳変数に影響されるソーステキスト部分のそれぞれの背景色を再設定し、ステップ 1903 で条件翻訳変数テーブルを再構築し、ステップ 1904 でソーステキスト 106 中の条件翻訳変数に影響されない共通ソーステキスト部分 1 と、翻訳条件が成立する条件翻訳領域をコンソール 111 に再表示し、以降のステップを繰り返す。

【0107】図 18 は、図 17 におけるステップ 1903 で実行される、条件翻訳変数テーブル作成処理 114 の処理の流れを説明するフローチャートである。

【0108】図 18 では、ソーステキスト 106 からソーステキストを 1 行づつ取り出し（ステップ 601）、ソーステキスト終端か判定し（ステップ 602）、ソーステキスト終端の場合、ソーステキスト 106 中に共通ソーステキスト部分 1 が存在するか判定し（ステップ 603）、存在する場合、条件翻訳変数テーブル 108 に、共通ソーステキスト部分 1 を示す特別なエントリを追加し（ステップ 604）、テキストエディタ使用者に共通ソーステキスト部分 1 として指定された背景色を追加した特別なエントリに設定する（ステップ 605）。

次にソーステキスト 106 中に共通ソーステキスト部分 2 が存在するか判定し（ステップ 606）、存在する場合、条件翻訳変数テーブル 108 に、共通ソーステキスト部分 2 を示す特別なエントリを追加し（ステップ 607）、テキストエディタ使用者に共通ソーステキスト部分 2 として指定された背景色を追加した特別なエントリに設定し（ステップ 608）、呼び出し元に処理を戻す。

【0109】ソーステキスト終端以外の場合、取り出したソーステキスト行の行頭が「#define」か判定し（ステップ 609）、行頭が「#define」の場合、#define 条件翻訳指示子から条件翻訳変数の名称と値を抽出し（ステップ 610）、条件翻訳変数テーブル 108 に新しいエントリを追加し（ステップ 611）、抽出した条件翻訳変数の名称と値と取り出した行の行番号と、この条件翻訳変数に影響されるソーステキスト部分としてテキストエディタ使用者に指定された背景色を、追加したエントリに設定し（ステップ 612）、ステップ 601 に戻り以降の処理を繰り返す。

【0110】行頭が「#define」以外の場合、行頭が「#if」かどうか判定し（ステップ 613）、行頭が「#if」以外の場合、ステップ 601 に戻り以降のステップを繰り返す。

【0111】行頭が「#if」の場合、#if 条件翻訳指示子の翻訳条件中に指定された条件翻訳変数の名称を抽出し（ステップ 614）、抽出した条件翻訳変数の名称が既に条件翻訳変数テーブル 108 に存在するか判定する（ステップ 615）。既に条件翻訳変数テーブル 108 に存在する場合、ステップ 601 に戻り以降のステップを繰り返す。

【0112】抽出した条件翻訳変数の名称が条件翻訳変数テーブル 108 に存在しない場合、条件翻訳変数テーブル 108 に新しいエントリを追加し（ステップ 616）、抽出した条件翻訳変数の名称と、この条件翻訳変数が未定義であることを示す値と、この条件翻訳変数に影響されるソーステキスト部分としてテキストエディタ使用者に指定された背景色を、追加したエントリに設定し（ステップ 617）、ステップ 601 に戻り以降の処理を繰り返す。

【0113】以上で説明した実施例によって、本発明の第一の実施例と同様の効果を持ちながら、本発明の第一の実施例により表示したソーステキストに対して、テキストエディタ使用者が、条件翻訳変数を容易に変更する手段を提供することで、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できる。更に本実施例によれば、ソーステキスト中に分散して記述されている任意の条件翻訳変数に対する参照・変更が容易になるという効果を持つ。

【0114】なお、以上で説明した実施例では、ソース

テキストウィンドウ 301 には、ソーステキスト中の条件翻訳指示子の翻訳条件を判定し、翻訳条件が成立する条件翻訳領域だけを表示するという表示方法であるが、本実施例における条件翻訳指示子の翻訳条件の判定および判定結果に基づいて条件翻訳領域の表示の有無を決定する処理の代わりに、ソーステキスト中の条件翻訳領域をアウトライン表示する処理を実行することによって、ソーステキストウィンドウ 301 に条件翻訳領域をアウトライン表示することができ、かつ条件翻訳変数ウィンドウ 302 に条件翻訳変数テーブル 108 の内容を表示し、テキストエディタ使用者はこのウィンドウに表示された条件翻訳変数テーブル 108 の内容を参照・変更するとともに、テキストエディタ使用者が条件翻訳変数テーブル 108 の内容を変更した場合、変更内容をソーステキスト 106 中の対応する #define 条件翻訳指示子に反映し、条件翻訳指示子の翻訳条件の再判定を行い、ソーステキストウィンドウ 301 の表示内容を更新するようにできることは言うまでもない。この方法と本実施例で説明した方法を選択するように構成することによって、テキストエディタ使用者は、ソーステキストウィンドウ 301 の表示内容を、翻訳条件が成立する条件翻訳領域だけを表示するか、条件翻訳領域をアウトライン表示するか、必要に応じて切替えることができる。

【0115】以上で説明した本発明の第一、第二及び第三の実施例を組み合わせることにより、第一の実施例によるソーステキスト表示処理と、第二の実施例によるソーステキスト表示処理を切替えることで、テキストエディタ使用者は、ソーステキスト全体を表示させる方法と、条件翻訳領域において翻訳条件が成立する条件翻訳領域だけを表示させる方法と、翻訳条件が成立するか否かにかかわらず任意の条件翻訳領域を表示させる方法とを、必要に応じて任意に選択することができるとともに、各ソーステキスト表示方法において、条件翻訳変数を変更し、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できるという効果を持つ。

【0116】なお、以上で説明した本発明による第一、第二及び第三の実施例では表 1 の条件翻訳指示子を使用して説明したが、プログラム言語 COBOL の次期規格では、図 19 のような条件翻訳指示子が規定される予定である。

【0117】本発明によるテキストエディタが図 19 の条件翻訳指示子を読み込む場合、「>>WHEN」から次の「>>WHEN」までの領域を #if ~ #endif と置き換えてから本発明の第一、第二及び第三の実施例の処理と同様の処理を行うことで、本発明による第一、第二及び第三の実施例と同様の効果を持つことができる。

【0118】なお本発明の第一、第二及び第三の実施例を組み合わせることにより、テキストエディタ使用者は、ソーステキスト全体を表示させる方法と、条件翻訳

領域において翻訳条件が成立する条件翻訳領域だけを表示させる方法と、翻訳条件が成立するか否かにかかわらず任意の条件翻訳領域を表示させる方法とを、必要に応じて任意に選択することができるとともに、各ソーステキスト表示方法において、条件翻訳変数を変更し、種々の翻訳条件の設定を可能にし、設定された翻訳条件に基づいて展開されたソーステキストを容易に確認できるという効果を持つ。

【0119】

【発明の効果】本発明によれば、従来技術で問題であった、条件翻訳領域が入れ子構造になっているような複雑な条件翻訳を含むソーステキストの場合に、翻訳条件に従って展開されたソーステキストを効率よく表示することができないという問題を軽減することができる。また、従来技術で問題であった、条件翻訳指示子を含むプログラムのソーステキストの場合に、ソーステキスト中の条件翻訳変数に影響されない共通なソーステキスト部分を容易に識別できないため、共通なソーステキスト部分に対する修正ミスが発生しやすいという問題を軽減することもできる。

【図面の簡単な説明】

【図1】本発明の第一の実施形態によるテキストエディタの構成例を示すブロック図である。

【図2】従来技術の問題点の例を説明する図である。

【図3】本発明によるテキストエディタの説明で共通に使用する、入力ファイルの例である。

【図4】本発明の第一の実施形態によるテキストエディタが動作しているコンソールの画面表示例である。

【図5】本発明の第一の実施形態によるテキストエディタのソーステキストウィンドウの表示例である。

【図6】本発明の第一の実施形態によるテキストエディタの処理の流れを示すフローチャートである。

【図7】本発明の第一の実施形態によるテキストエディタの背景色設定処理の処理の流れを示すフローチャートである。

【図8】本発明の第一の実施形態によるテキストエディタの条件成立翻訳領域表示処理の処理の流れを示すフローチャートである。

【図9】本発明の第二の実施形態によるテキストエディタの構成例を示すブロック図である。

【図10】本発明の第二の実施形態によるテキストエ

ィタのソーステキストウィンドウの表示例である。

【図11】本発明の第二の実施形態によるテキストエディタの、図10における非展開状態のアウトライン表示領域に対して、展開するよう操作を行った場合の表示例である。

【図12】本発明の第二の実施形態によるテキストエディタの処理の流れを示すフローチャートである。

【図13】本発明の第二の実施形態によるテキストエディタのアウトライン表示処理の処理の流れを示すフローチャートである。

【図14】本発明の第三の実施形態によるテキストエディタの構成例を示すブロック図である。

【図15】本発明の第三の実施形態によるテキストエディタのコンソールの表示画面例である。

【図16】本発明の第三の実施形態によるテキストエディタの条件翻訳変数ウィンドウの表示例である。

【図17】本発明の第三の実施形態によるテキストエディタの処理の流れを示すフローチャートである。

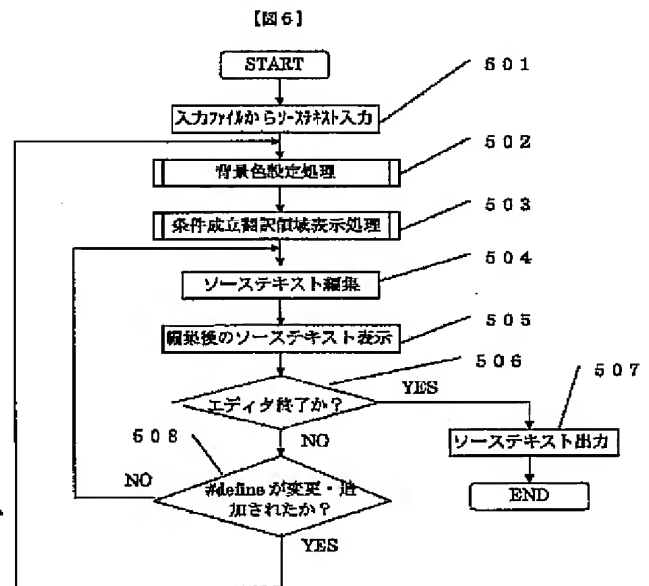
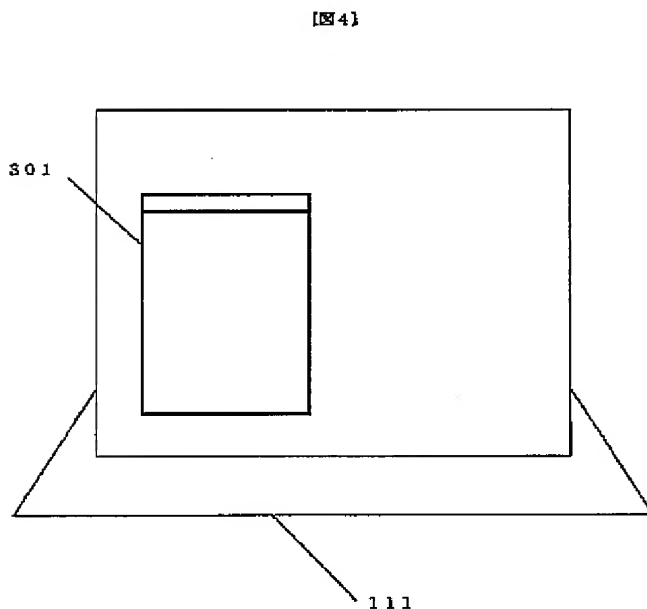
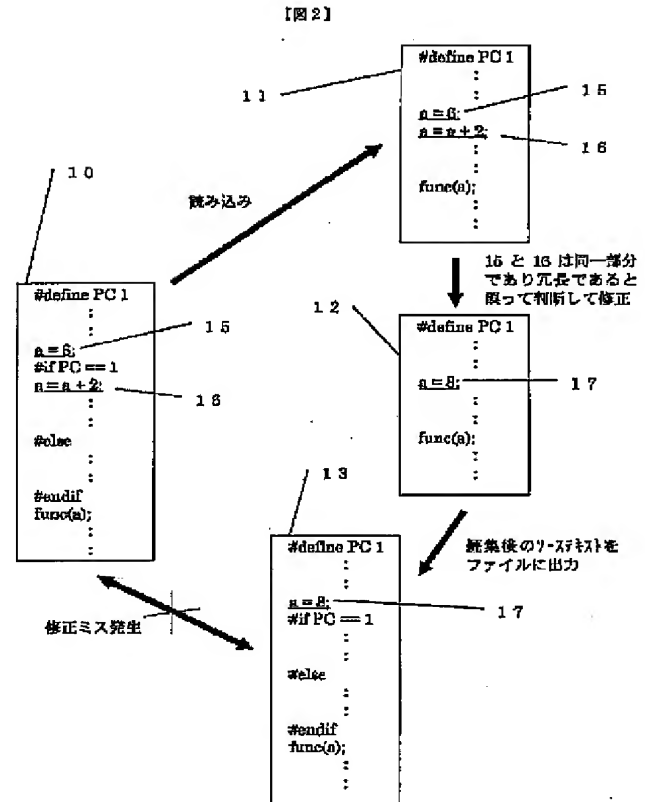
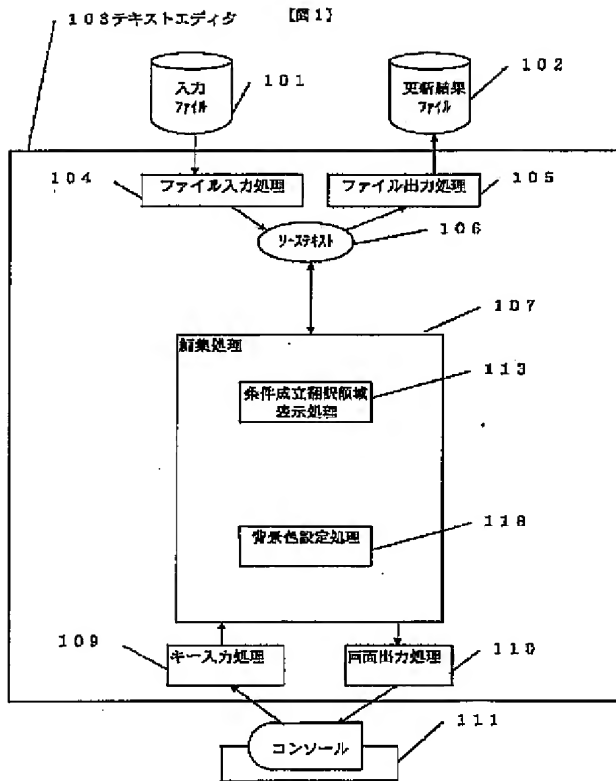
【図18】本発明の第三の実施形態によるテキストエディタの条件翻訳変数テーブル作成処理の処理の流れを示すフローチャートである。

【図19】プログラム言語COBOLの次期規格における条件翻訳指示子の例である。

【図20】条件翻訳指示子の文法と意味を示す。

【符号の説明】

- 101 入力ファイル
- 102 更新結果ファイル
- 103 テキストエディタ
- 104 ファイル入力処理
- 105 ファイル出力処理
- 106 ソーステキスト
- 107 編集処理
- 108 条件翻訳変数テーブル
- 109 キー入力処理
- 110 画面出力処理
- 111 コンソール
- 113 条件成立翻訳領域表示処理
- 114 条件翻訳変数テーブル作成処理
- 115 条件翻訳変数テーブル表示制御処理
- 117 アウトライン表示処理
- 118 背景色設定処理



【図3】

【図3】

```

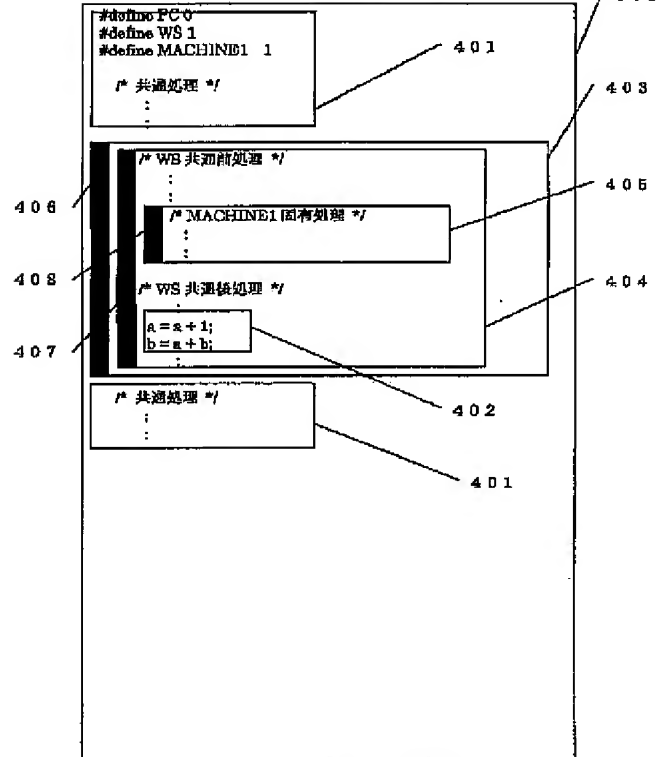
#define PC 0
#define WS 1
#define MACHINE1 1

/* 共通処理 */
:
:
#if PC == 1
/* PC固有処理 */
:
:
a = a + 1;
b = a + b;
:
:
#else
/* WS固有処理 */
:
:
a = a + 1;
b = a + b;
:
:
#endif
/* WS 共通後処理 */
:
:
a = a + 1;
b = a + b;
:
:
/* PC/WS 以外の処理 */
:
:
#endif
#endif
/* 共通処理 */
:
:

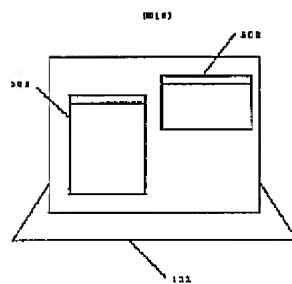
```

【図5】

【図5】



【図15】



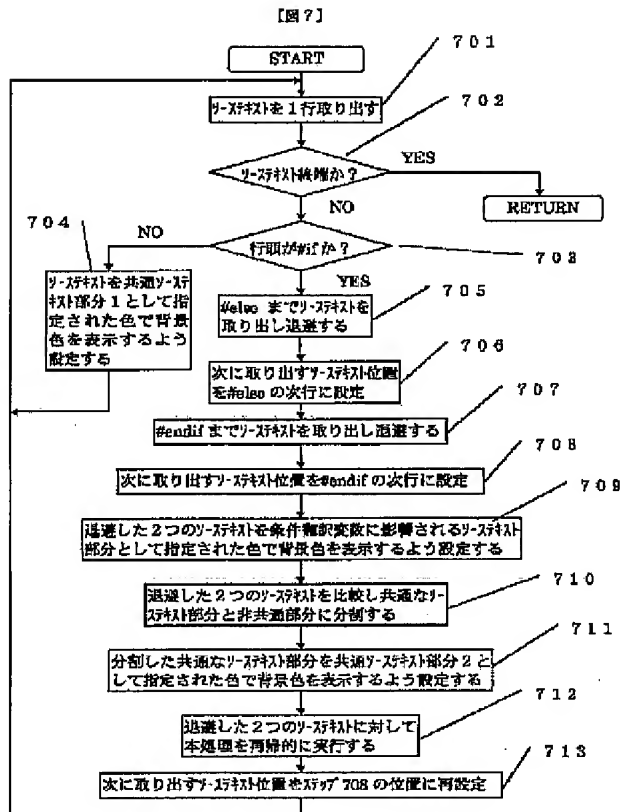
【図16】

【図16】

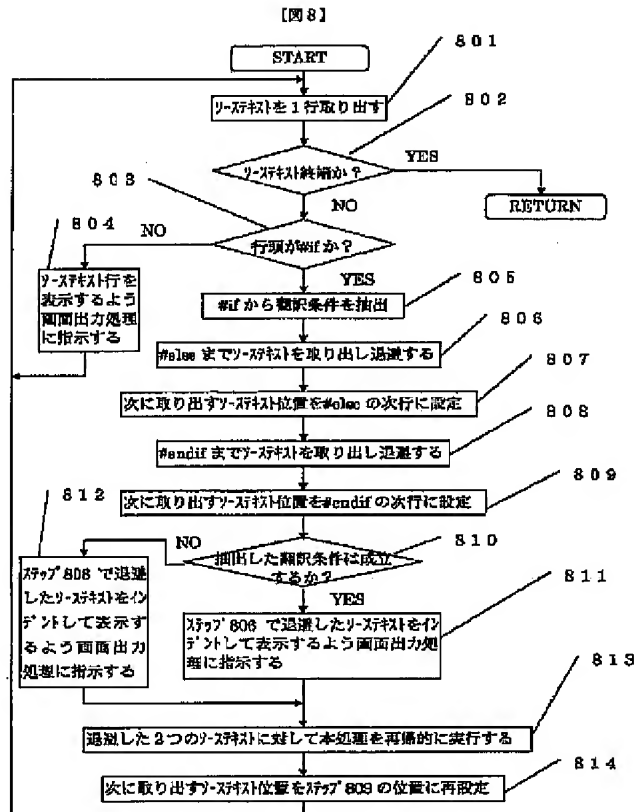
項番	名称	値	行番号	背景色
1	PC	0	1	赤
2	WS	1	2	緑
3	MACHINE1	1	3	青
4	共通リソース部分1	—	—	白
5	共通リソース部分2	—	—	黒

302

【図7】



【図8】



【図19】

【図19】

```

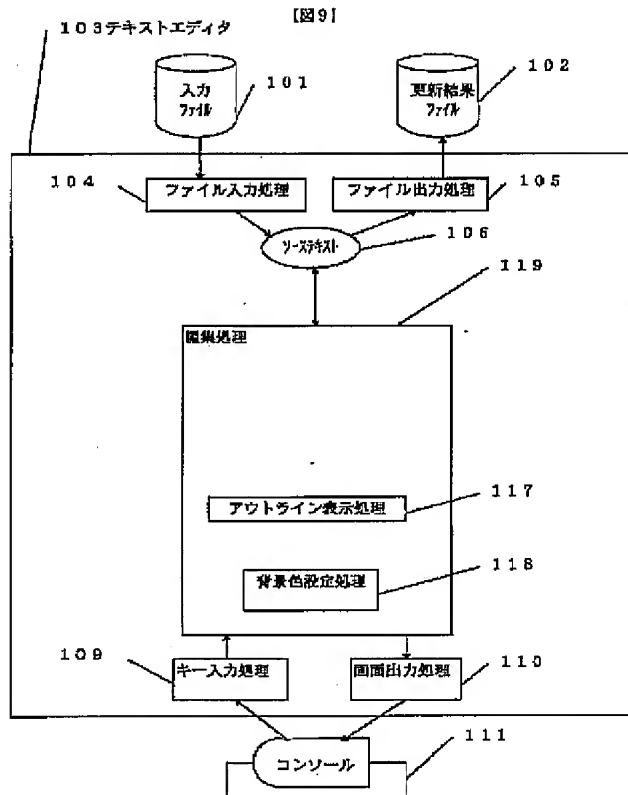
>>EVALUATE  翻訳条件
>>WHEN  翻訳条件
    実行文
    :
>>WHEN  翻訳条件
    実行文
    :
>>WHEN OTHER
    実行文
    :
>>END-EVALUATE
  
```

【図20】

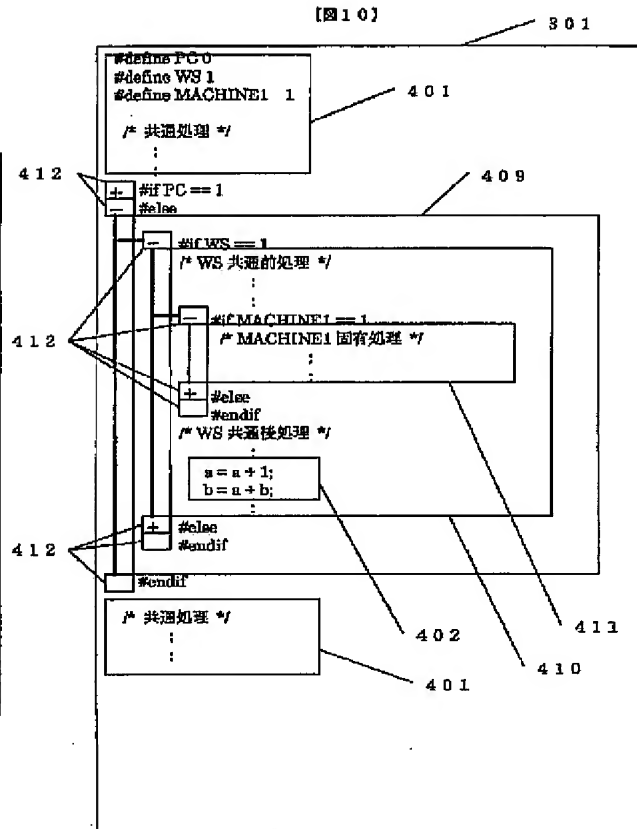
【図20】

指示子	文法	説明
#define	#define 名称 値	#if 指示子の翻訳条件式で使用する条件翻訳変数を定義する。
#if	#if 翻訳条件式	翻訳条件式が成立する場合、対応する#elseが存在すれば#elseまでの範囲のソーステキストを、対応する#elseが存在しない場合は#endifまでの範囲のソーステキストを読み込む。
#else	#else	翻訳条件式が成立しない場合、対応する#elseが存在すれば#else以降#endifまでの範囲のソーステキストを読み込み、対応する#elseが存在しない場合は#endifまでの範囲のソーステキストを読み飛ばす。
#endif	#endif	対応する#if指示子の翻訳条件式が成立しない場合に読み込むソーステキスト範囲の開始を示す。
		条件翻訳のソーステキスト範囲の終了を示す。

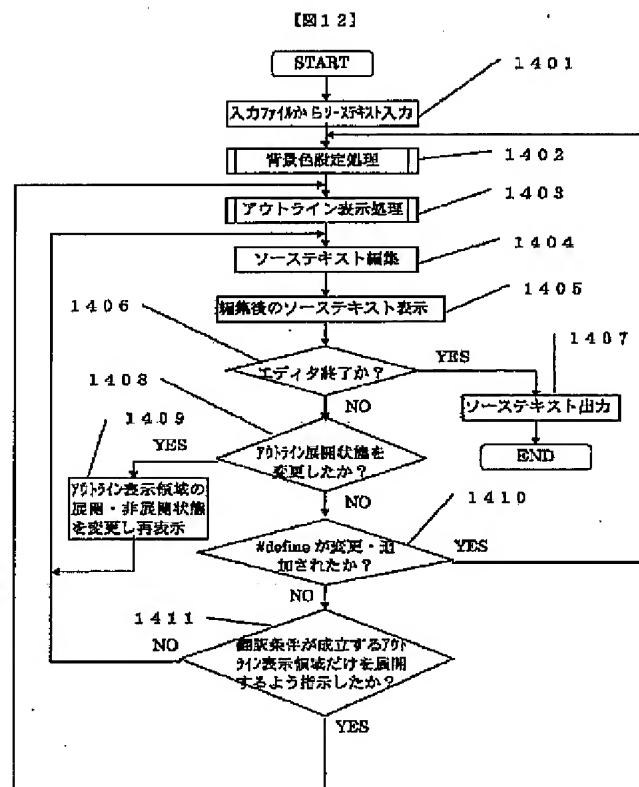
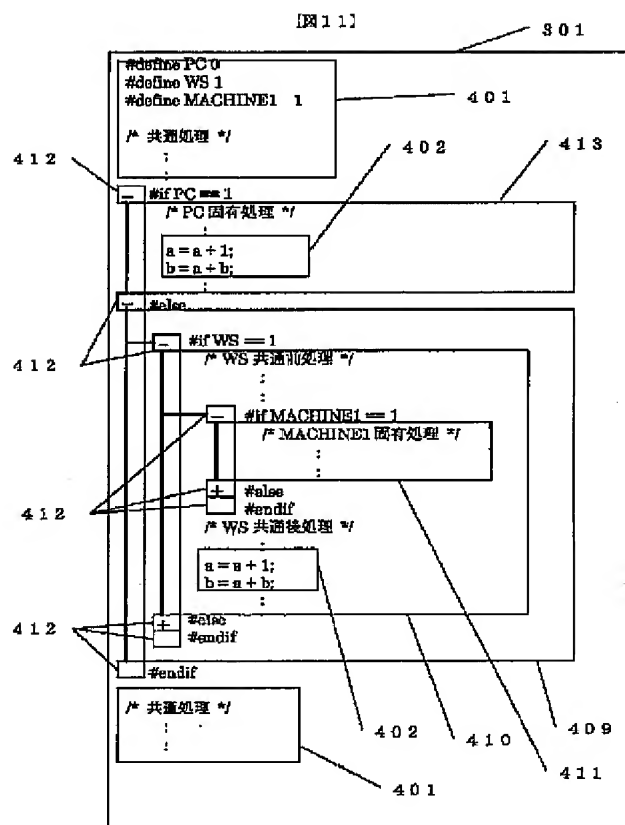
【図9】



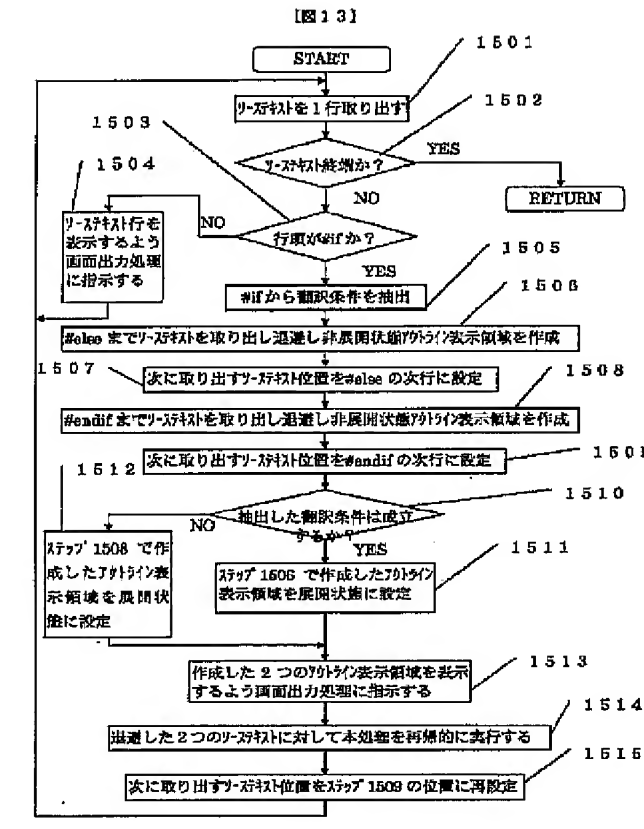
【図10】



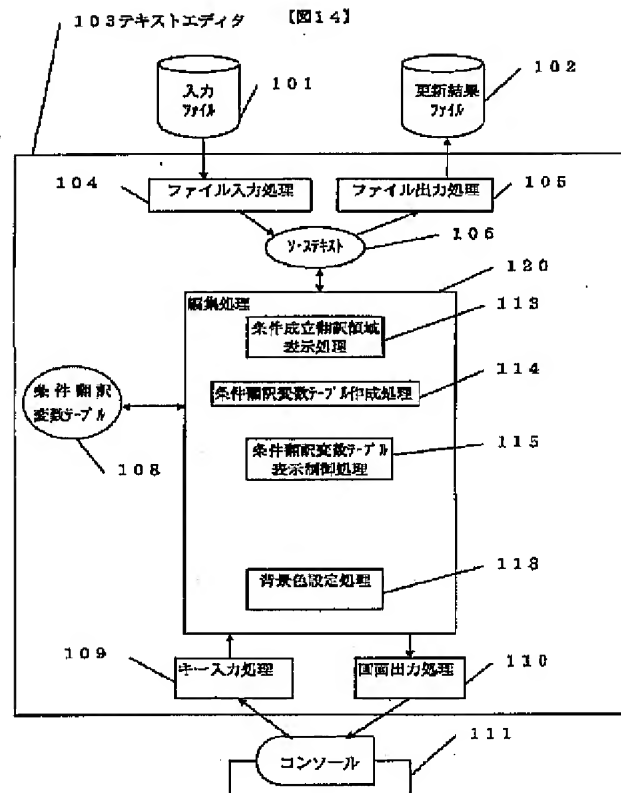
【图 12】



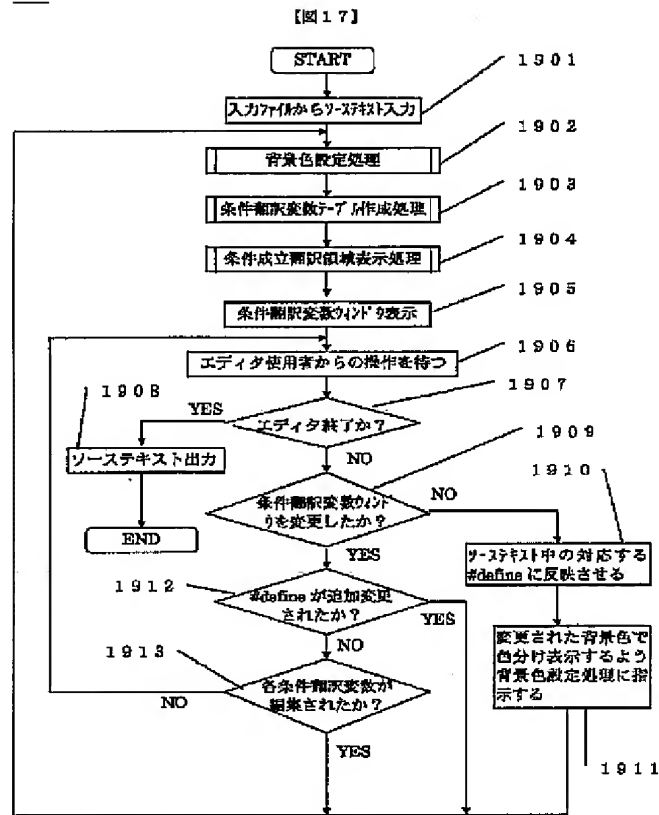
【図13】



【図14】



【図17】



[X 18]

